Vol.4,No.1, July, 2015
134 - 140,ISSN:0975-5446 [VNSGU JOURNAL OF SCIENCE AND TECHNOLOGY]

Clustering NSE scripts using *K-means* algorithm with *KD-tree*

RANA P. C.
Asst. Prof.
Department of Computer Science
VNSGU, Surat
pcrana@gmail.com

MORENA R. D.
Professor
Department of Computer Science
VNSGU, Surat
rdmorena@rediffmail.com

Abstract
We have applied *K-means* algorithm to form clusters of scripts for the Indian Stock Market, based on their end of day (EOD) data. The *K-Means* clustering algorithm, at its first step, requires the centroids (also termed as seeds) for determining the clusters. Using *KD-tree*, we derived this initialization set of centroids, and found that more converged clusters are formed. The dataset is based upon the percentage change of the end-of-day (EOD) data of scripts listed on the National Stock Exchange (NSE, India).

Keywords : Clustering, *K-means*, *KD-tree*, cluster, centroid, seed, convergence

1. Introduction

Clustering has been the basis of many knowledge discovery tasks like machine learning, statistics, data mining, pattern recognition etc. There are two main branches of clustering; Hierarchical and Partitioned [2]. In this paper we concentrate on partitioned clustering, and in particular, a popular partitioning clustering method called *K-means* clustering.

Clustering is a data mining technique used to place data elements into related groups without prior knowledge of the clusters to be determined. The procedure follows a simple way to classify a given data set through a certain number of clusters fixed a priori. The method is called *K-means* since each of the K clusters is represented by the mean of the objects, called as centroids, within it. In K-means, one needs to partition a given set of data points into a number of distinct groups, termed clusters. In this paper, by data we will refer to n data points spanning m dimensional space and we denote K as the number of clusters the data will be partitioned into.

Algorithm 1 - K-means Algorithm

1. Initialize K center locations (c_1,...., c_K).
2. Assign each x_i to its nearest cluster centre c_i.
3. Update each cluster centre c_i as the mean of all x_i that have been assigned as closest to it.

4. Calculate Distortion D = $\sum_{1=1}^{n} [\min_{(j=1..k)} d(x_j, c_i)]^2$
5. If the value of D has converged, then return (c_1,...., c_K); else go to Step 2.

The best applied algorithm for clustering tasks, in particular for stock market, is the *K-means* algorithm [9]. The *K-means* algorithm attempts to find the cluster centers *(c_1,...., c_K)*, such that the sum of the square of distances (termed as Distortion, *D*) of each data point (x_i) to its nearest cluster centre (c_i), is minimized, where *d* is the distance function. Typically *d* is chosen as the Euclidean distance. A pseudo-code for the *K-means* algorithm is shown in Algorithm 1.

K-means algorithm is a local optimization strategy and is sensitive to the choice of the initial positions of the cluster centers *(c_1,...., c_K)*. These initial centre locations are known as the centroids (or seeds) for the *K-means* algorithm. The most commonly used initialization technique has been the repeated runs of Forgy's algorithm and choosing the final clustering, that returns the smallest value of the Distortion [1]. However, while repeated runs of the Forgy's method appears to be the de facto approach to initialization of the K-means algorithm, many other techniques have been proposed like McQueen Approach [2], Simple Cluster Seeking Method, Binary Splitting, Optimal Seed Selection method using genetic programming, Direct Search Binary Splitting, KKZ algorithm, Global *K-means*, Cluster Center Initialization Method using Density-Based Multi-scale Data Condensation, *KD-Tree* etc. [1][2].

We employ the cluster analysis on the equity scripts of the National Stock Exchange (NSE) and determine the clusters, using *KD-Tree*, based upon the percentage change of their end-of-day (EOD) data. Past studies have focused on simply deriving the clusters, simply by selecting randomly the centroids from the dataset [9]; but better convergence was anticipated for the resultant clusters. Also, in *K-means* clustering technique, the resultant clusters are much dependent on the initial seeds or centroids chosen [6]. Hence, we felt a need to rethink upon the selection of the initial seeds; where we opted to apply *KD-Tree* [1][2][3]. These generated centroids can then be used for applying *K*-means on our scripts dataset so as they converge to the correct or near clusters. Once stocks are grouped by cluster analysis, an informed investor can use the output in his interests. He will, for instance, look for same-return stocks and then choose to minimize risks [4].
The rest of the paper is organized as follows. Section 2 discusses the related work in this area, Section 3 discusses about the methodology, where in it focuses upon data procurement, cleansing of the EOD data, and modeling. In Section 4, we perform the testing and infer the results of the total scripts falling under the clusters, and finally conclude in Section 5.

2. Related Work

The trends of the global stock markets are entirely different in different stock markets, as mentioned by Todd Wittman[3]. Past work is limited to only for a specific vertical group of industries, viz. chemicals, biotechnology & drugs, retail, healthcare and pharma. Very little work has been done on verifying which measures are most suitable for mining of a given class of data sets, as mentioned by Martin Gavrilov, Dragomir Anguelov, Piotr Indyk, Rajeev Motwani [7]. S.R. Nanda, B. Mahanty, M.K. Tiwari [9] have proposed Kmeans cluster analysis for portfolio management, which select stocks from the clusters to build a portfolio, minimizing portfolio risk and compare the returns with that of the benchmark index, i.e. Sensex. Also, similar work has been carried out by Ruizhong Wang [8], where he used cluster analysis for the constituent stocks on China Shanghai 180, taking the ability of profitability, capital expansion, asset management, growth and solvency certificate of the listed companies.
In this paper, our focus is to optimize space using *KD-Tree* to determine the initial centroids, which can then be used by *K-means* for better convergence of clusters. The number of seeds to be taken initially is a sensitive issue [6]. Stephen J. Redmond & Conor Henenghan

mentioned that "one of the earliest references to initializing the *K-means* algorithm was by Forgy in 1965" [1]. They quoted that, "Forgy simply suggested that K instances should be chosen from the database at random and used them as the seeds". This approach takes advantage of the fact that if we choose points randomly, we are more likely to choose a point near a cluster centre, by virtue of the fact that this is where the highest density of points are located. However, there is no guarantee that we will not choose two seeds near the centre of the same cluster, or that we will not choose some poorly situated outlying point. In fact, repeated runs of this method, is presently the standard method of initializing the *K-means* algorithm.

Furthermore, if the number of clusters (and hence the number of centroids) increases, it would be difficult to determine the exact cluster that deserve to contain the resultant value. Hence, for better convergence of the sample, we use *KD-Tree* for initializing the *K-means* centroids or seeds.

3. Methodology

Data mining has been applied on the relative pricing of the stocks on their past data. Our approach will be first to determine the classes and identify the class labels and group them to determine the relativity amongst the stock groups. We have determined some good quality clusters in which the intra-class similarity is high and the inter-class similarity is low, with an ability to determine the association between the stocks. Before applying *K-means*, we need to get the seeds or initial centroids. By combining K-Means and hierarchical clustering, it can improve the performance and make the result more accurate [10].
We hereby illustrate the whole data mining process using the standard methodology in the following subsections.

3.1 Data Procurement

A stock market or equity market is a public entity for the trading of company shares at an agreed price listed on the most popular stock exchanges of the country. The study is based on the EOD data of equities available for the period of 6 years for all the transactions at NSE, i.e. 2006 to 2011. The data is in the form of a *csv* (comma separated value) file, which contains the data of the script name, script type, opening price, high price, low price, previous day closing price, quantity traded, and trade-date.

3.2 Data Cleansing

There were 1744065 records in the *csv*; which we separated into year wise tables and migrated the data into MS-SQL. We applied data cleansing on the file, where we filtered out the scripts of type EQ (equity) first, resulting into 1691248 records. Thereafter, we separated the data year-wise into separate MS Excel workbooks, and then imported the data into MS-SQL. Then, we sorted our data on the order of the stock name (ascending) and transaction dates (descending). Further, we deleted those records which had null values in any of the fields. Also, we removed the NSE category records which were of type "CNX%, since they are outliers of our data set. As the distance measure is sensitive to noise and outliers, it may pose an issue for financial data; which may occur due to an abrupt jump in stock prices following a merger, bankruptcy, or scandal [5].

For our study, we aim at deriving the best centroids so as to form well converged clusters on an individual day. By simply querying the data on a particular transaction date, we get the dataset to be used for our clustering purpose.

3.3 Modeling
KD-trees

The *KD-tree* is a top-down hierarchical scheme for partitioning data. Consider a set of n points, $(x_1...x_n)$ occupying *m* dimensional space. Each point x_i has been associated with it *m* co-ordinates $(x_{i1}, x_{i2}, ..., x_{im})$. There exists a bounding box, or bucket, which contains all data points and whose extrema are determined by the maximum and minimum co-ordinate values of the data points in each dimension. The data is then partitioned into two sub-buckets by splitting the data along the longest dimension of the parent bucket, which we denote as m_{max}. This partitioning process thereafter can be recursively repeated on each sub- bucket until a leaf bucket (denoted as *L*) is created; at which point no further partitioning will be performed on that bucket. *L* is a bucket which fulfills a certain requirement, such as, it only contains 1 data point or contains less than 10 data points [11]. Eventually, after enough splitting of buckets, all buckets will be leaf buckets and the partitioning process will terminate. With the *KD-Tree* approach, at most n buckets will need to be dealt with as the space has been more intelligently divided.
We start by creating a *KD-tree*, stipulating that a leaf bucket is that which, arbitrarily, contains 10 buckets for each cluster. We count the number of leaf buckets created. We calculate the volume of each leaf bucket, and count the number of points it bounds. We then calculate the density of each leaf bucket. We must associate each leaf bucket, or more exactly the density of each leaf bucket, with a point in the space. We choose this point to be the mean of the data points contained within the leaf bucket. So, we now have a list of q points $(m_1, ...,m_q)$ in the space and the corresponding estimates of the density of the data at each point $(p_1, ..., p_q)$.

To choose the initial seeds for the *K-means* algorithm, we wish to use this density information. We aim to choose *K* leaf bucket locations m_j, from *q* possibilities, that are separated by a reasonable distance and have a large density. We choose the first seed to be the leaf bucket with highest density. To choose the second seed we calculated, for every remaining leaf bucket centroid, the value g_j being the distance of m_j from the first seed location using the Euclidean distance, multiplied by the density of the leaf bucket, The second seed is then chosen as the point, m_j , with the maximum value of *g*. The idea is that the further away a leaf bucket is from an existing seed, and the larger its density, the more likely a candidate it is to be an initial centre location. Similarly, the third centre is chosen by computing the distance of each leaf bucket centroid from its nearest seed location multiplied by the density of the leaf bucket itself.

To use *K-means*, we first need to arrive upon a data set which depicts uniformity amongst all the records, regardless to their open, high, low, close or qty values. Hence, we calculated the % change in the difference between the LTP (last traded price) of the script on a particular day and previous day. This was then exported to a *csv* file, and likewise, we derived a *csv* for each year. The *EOD NSE* data exists for 1453 scripts of type EQ; which we are interested in, on a daily basis. First, we applied *KD-Tree* to determine the initial seeds or centroids as discussed above and got the centroids for K=10 as shown in Table 1 from the data as shown in Table 3 (only the first few records are shown).

Table 1- Centroids determined using KD-tree for K=10

AXISBANK	-4	-3.26	-4.67	-4.97	61.76
DLF	-5.76	-5.18	-3.13	-2.49	0.16
FORTIS	-2.22	-1.65	-1.46	-1.15	-23.02
GSPL	-2.35	-	-3.4	-2.18	84.35

		2.44			
IDFC	-5.86	-6.1	-3.65	-3.02	-23.38
JPASSOCIAT	-1.14	-0.19	-3.72	-3.38	38.63
KOUTONS	-4.88	-3.68	-4.29	-2.38	37.03
POWERGRID	-0.48	1.05	-0.39	1.83	131.76
TATASTEEL	-8.69	-8.72	-4.72	-3.97	2.89
WIPRO	1.27	1.79	0.31	1.49	220.39

The initial sample data on the transaction date basis is shown in Table 2.

Table 2 – NSE-EOD sample Dataset

Sname	**Open**	**high**	**low**	**Close**	**Qty**	**Tdate**
20MICRONS	56.95	57.5	56	56.3	56852	2011-11-29
3IINFOTECH	19.2	20	18.85	19.1	818481	2011-11-29
3MINDIA	3810	3886	3750	3814.8	188	2011-11-29
AARTIIND	48.2	49	47.1	47.6	7426	2011-11-29
ABAN	357.9	364.5	350	353.35	883139	2011-11-29
ABB	610	620	596	599.9	116529	2011-11-29
ABGSHIP	404.95	408.8	400	405.35	79968	2011-11-29
ABIRLANUVO	901.95	915	892.25	901.65	183459	2011-11-29
ACC	1186.15	1192	1148	1160.9	180659	2011-11-29
ADANIENT	320	326.7	300.65	305.7	895255	2011-11-29
ADANIPOWER	73.55	74.4	71.65	72	377001	2011-11-29

The percentage differences between the Open, High, Low, Close, Qty of a particular date with the previous date values (sample data) is shown in Table 3.

Table 3 – Dataset calculated for the formation of clusters

Script Name	**Open % change**	**High% change**	**Low% change**	**Close (LTP) % change**	**Qty % change**	**Transaction Date**	**Previous Tdate**
20MICRONS	-2.99	-1.86	-0.46	-0.15	1.97	2011-11-29	2011-11-28
3IINFOTECH	-1.62	-2.57	-4.12	-3.49	24.14	2011-11-29	2011-11-28
3MINDIA	-1.35	-1.65	-4.07	-3.88	28	2011-11-29	2011-11-28
AARTIIND	-7.36	-6.54	-8.22	-4.67	105.16	2011-11-29	2011-11-28
ABAN	-1.38	-2.58	0.27	1.12	-52.84	2011-11-29	2011-11-28
ABB	11.45	9.97	-3.43	-0.67	66.54	2011-11-29	2011-11-28
ABGSHIP	-1.04	-1.23	-1.28	1.06	49.35	2011-11-29	2011-11-28
ABIRLANUVO	-2.74	0.93	-9.31	-0.68	-73.76	2011-11-29	2011-11-28
ACC	-1.15	-4.17	-2.59	-1.6	-52.62	2011-11-29	2011-11-28
ADANIENT	-0.31	-2.08	-2.39	-0.72	-66.11	2011-11-29	2011-11-28
ADANIPOWER	-1.61	1.62	-0.07	0.02	907.12	2011-11-29	2011-11-28

We implemented the *K-means* algorithm using 'C' programming language. The above Table 3 (data set) and Table 1 (Centroids) are taken as input, and the program generates the output file showing the cluster numbers and the total scripts falling under the respective clusters.

The *K-means* algorithm converges after there is no movement amongst the scripts across the clusters.

4. Results and Testing

We repeated the same algorithm for another 50 days from the records set, and found the following results, for 1453 scripts we chose for studying the associative trend as follows:
Table 4

Cluster No.	No. of scripts with K=10 using Forgy's method	No. of scripts with K=10 using KD-tree
1	135	127
2	134	138
3	203	201
4	187	183
5	212	202
6	104	114
7	89	102
8	178	183
9	118	102
10	93	101

We studied the clusters and the scripts falling under them. Upon comparison of the scripts clustered with centroids of the default Forgy's method, we found more accurate co-relations between scripts when we clustered using centroids derived using *KD-tree*. The scripts falling under a particular cluster, heavily depends upon the seed or centroid values taken at random during the initial steps of the *K-means* algorithm. Future work needs to access the effect on these clusters, using varying centroids viz. changing *K* repeatedly for determining co-relations between scripts.

6. Conclusion

There is a lot of risk trading in the stock market with limited information and uncertainty. The study reveals that to determine the associative relationship of scripts clustered together exhibiting similar behavior on daily trading pattern, were more accurate when the centroids are determined using *KD-tree*. Using our methodology, one can determine the clusters of stocks that exhibit similarities and would ascertain the investor or trader to initiate trade or increase the chances of maximizing profits.

Acknowledgement

We acknowledge UGC for Special Assistance Program (SAP) for Natural Language Processing and Data Mining (file number is F.3-48/2011) under which this work has been done.

References

1. "A method for initializing the K-means clustering algorithm using kd-trees", Stephen J. Redmond & Conor Henenghan, Department of Electronic Engineering, University College Dublin, Belfield, Dublin, Ireland, Preprint submitted to Elsevier Science 21st October 2005.
2. "An Efficient k-Means Clustering Algorithm: Analysis and Implementation" by Tapas Kanungo, Senior Member, IEEE, David M. Mount, Member, IEEE, Nathan S.

Netanyahu, Member, IEEE, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu, Senior Member, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24. No.7, pp. 881-892, 2002

3. "Selection of K in K-means clustering" by D T Pham, S S Dimov, and C D Nguyen, Manufacturing Engineering Centre, Cardiff University, Cardiff, UK published in 2005 Proceedings of IMechE Vol. 219 Part C: J. Mechanical Engineering Science, page nos. 103-119.
4. Da Costa, Jr, Newton, Jefferson Cunha, and Sergio Da Silva, (2005) "Stock selection based on cluster analysis." Economics Bulletin, Vol. 13, No. 1 pp. 1−9, October 2005.
5. "Time-Series Clustering and Association Analysis of Financial Data" by Todd Wittman, CS 8980 Project December 15, 2002, unpublished.
6. "Hierarchical K-means: an algorithm for centroids initialization for K-means", by Kohei Arai and Ali Ridho Barakbah, Reports of the Faculty of Science and Engineering, Saga University, Vol. 36, No.1, 2007.
7. Martin Gavrilov , Dragomir Anguelov , Piotr Indyk , Rajeev Motwani, Mining the Stock Market: Which Measure is Best, 6 th ACM Int'l Conference on Knowledge Discovery and Data Mining, Stanford CA, 2000.
8. "Stock Selection Based on Data Clustering Method" by Ruizhong Wang, Information Engineering School, Tianjin University of Commerce, Tianjin, China published in 2011 IEEE Seventh International Conference on Computational Intelligence and Security.
9. Elsevier's Expert Systems with Applications Volume 37, Issue 12, December 2010, Pages 8793–8798, "Clustering Indian stock market data for portfolio management" by S.R. Nanda, B. Mahanty, M.K. Tiwari, Department of Industrial Engineering and Management, Indian Institute of Technology, Kharagpur, West Bengal, India.
10. J. Chen, Russell KH Ching and Yi-Shen Lin, "An Extended Study of the K – Means Algorithm for Data Clustering and Its Application", Journal of the Operational Research Society, Vol. 55, pp. 976 - 987, 2004.
11. "A Dynamic Linkage Clustering using KD-Tree" by Shadi Abudalfa and Mohammad Mikki published in The International Arab Journal of Information Technology, Vol. 10, No. 3, May 2013 pp 283-289